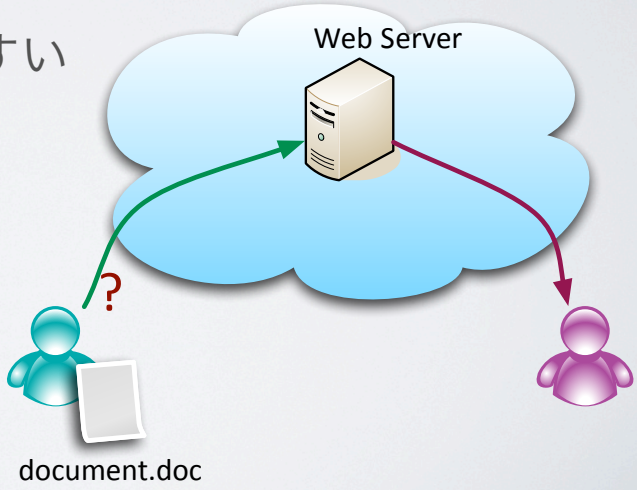
セッション毎に IPv6 アドレスを変動させる安全な通信方式とそのアドレス更新手法

西田　和生（大阪大学）
阿多　信吾（大阪市立大学）
北村　浩（NEC/電気通信大学）
村田　正幸（大阪大学）

やりたいこと（Web 版）

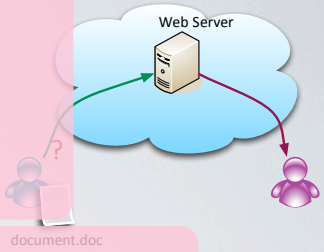
- **あるファイルを特定の他人と共有したい**
  - ただし不特定多数には存在を知られたくない
- **どこかの個人用 Web サイトを使う**
- **なるべく面倒なことはやりたくない**
  - わかりやすい＆やりやすい



Web サーバを使った共有

- Web サイトに認証を導入
  - ユーザ名とパスワードを登録
  - 認証ユーザのみダウンロードを許可

認証メカニズム？ログインページ？
ユーザ名とパスワード？



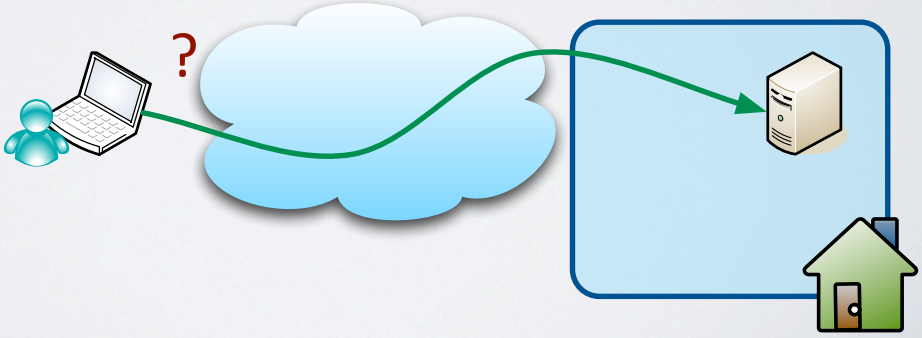
- ファイルを暗号化
  - 復号のためのパスワードを知らないと読めない

暗号・復号ソフトのインストール？

- **ファイル名を類推困難に**
  - document.doc → 0b2173c7a9ab63e8a31576f941bc1185028e6017.doc

やりたいこと (IP 版)

- **あるノードとインターネットを経由して通信したい**
  - ただし不特定多数には存在を知られたくない
- **自宅のネットワークにあるノードをサーバに**
- **なるべく面倒なことはやりたくない**
  - わかりやすい＆やりやすい



外部からの接続

- ファイアウォール・VPN の導入
  - ユーザ毎に VPN ルータを配布
  - 認証 OK のユーザのみ社内ネットワークへの接続を許可
- 通信路を暗号化
  - IPSec, MPPE など
- **IP アドレスを類推困難に**
  - 2001:DB8::1 → 2001:DB8::2106:5fb8:49cd:a1f6

VPN ルータ？フィルタリング設定？

ユーザ認証？

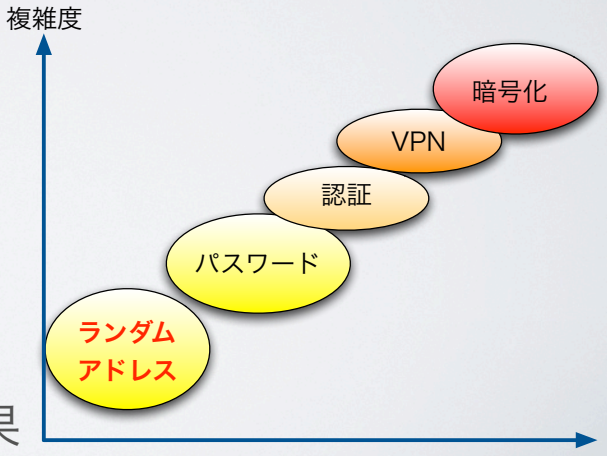
暗号化設定？

研究の目的

- **簡単な方法で安全・安心な通信を実現したい**
  - 「実感できるセキュリティ」の提供
- **通信セッションごとに変動するアドレスを使用した通信アーキテクチャ**
  - IPv6 アドレス自体を「鍵情報」として使用
  - 類推困難なアドレス生成系列
  - 再利用を禁止することによる安全性の向上
  - 通信ごとのアドレス共有手順が不要
    - エンドノードの自律的な同期によるアドレス共有

「実感できるセキュリティ」の実現

- **安全性と複雑度はトレードオフの関係**
  - 安全性を高めるほど複雑なしくみに
  - すべての通信に高い安全性は必要？
  - セキュリティに関する高い知識と技術
- **使いやすいしくみ**
  - 安全性は完璧ではないが容易に実現可能
  - しくみが簡単
  - **誰もが理解できる**
  - 組み合わせによる相乗効果

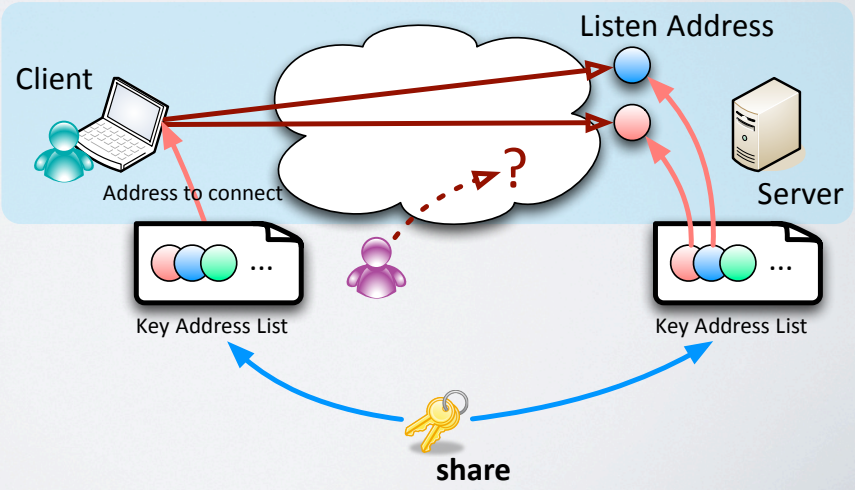


変動する IPv6 アドレスを用いた安全性向上

- **「実感できるセキュリティ」の実現**
  - 「知らない人にアドレスを教えない」はシンプルだが実効性が高い
  - 特にアドレス空間が広い IPv6 で有効（64ビット）
  - 使われるにつれて外部に伝わる可能性が上昇
    → 一度使用されたアドレスは再利用しない
  - 類推困難な情報を使用すれば安全性がさらに向上
- **「実感できるセキュリティ」による安心感の提供**
  - ディジタル・ディバイドの軽減
  - トラブルシューティングを容易に

提案方式概要

- **通信セッションごとに変動する IP アドレス**
  - 「専用」のアドレスにより、再利用を不可能に
- **IPv6 アドレス自体を「鍵」として使用**
  - 類推できないアドレスを使うことで外部からの接続を困難に

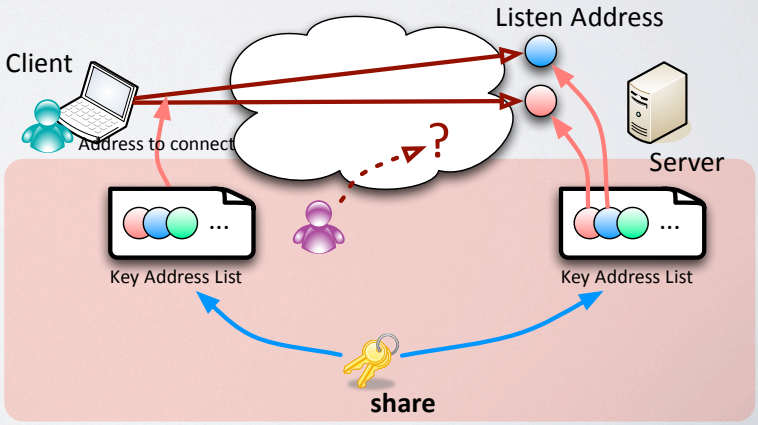


「鍵アドレス」の共有

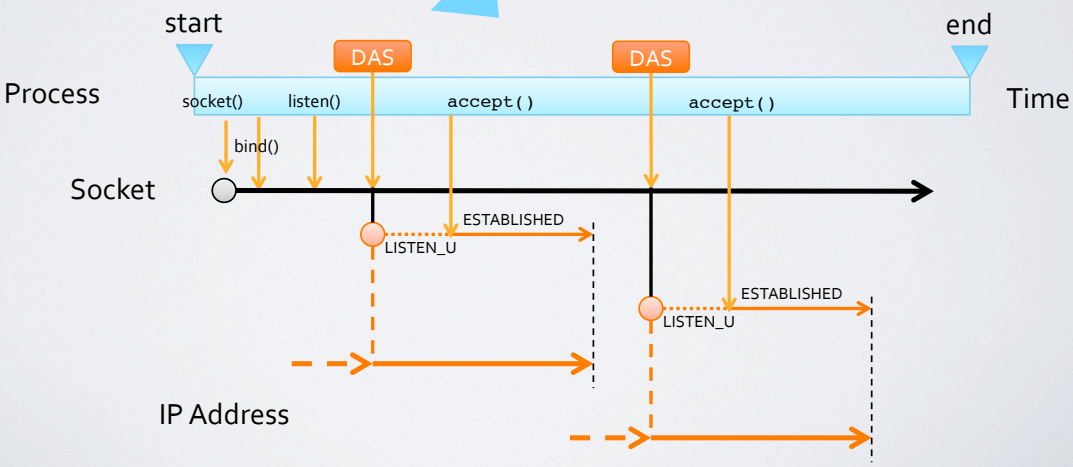
- **どうやってサーバの待ち受けアドレスを取得するか？**
  - 毎回サーバに問い合わせ
    - 問い合わせの非効率性
    - 問い合わせサーバの存在を隠匿できない
  - ブローカーによる仲介
    - しくみが複雑化
    - エンドノードで完結しない
- **エンドノードだけで自律的に鍵アドレスを共有**
  - 通信セッションごとにアドレスを問い合わせず、自ノードだけで解決

鍵アドレスリストによる自律型アドレス取得

- **サーバ・クライアント間で「鍵アドレスリスト」を生成し共有**
  - 実際には共有パスフレーズにより動的に生成
- **クライアントからの接続状況に応じた同期制御**
  - 通信セッションの状態による同期
  - 時刻による再同期

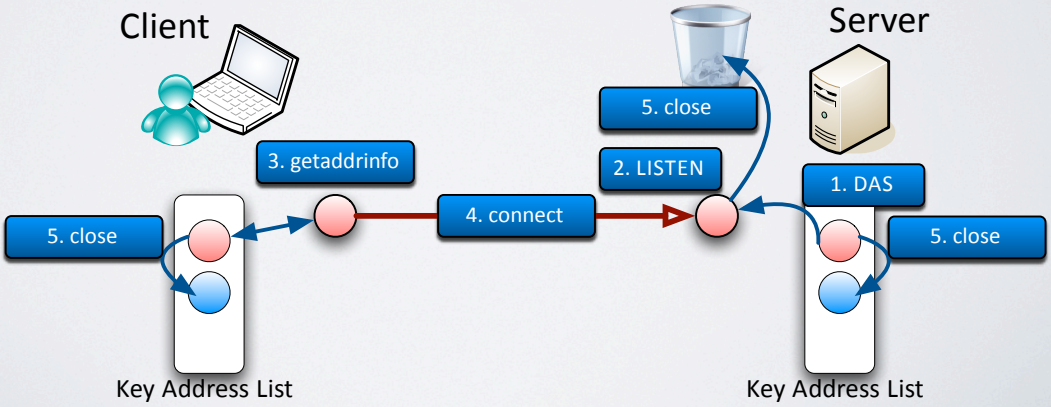


待ち受けアドレスとソケットの関係



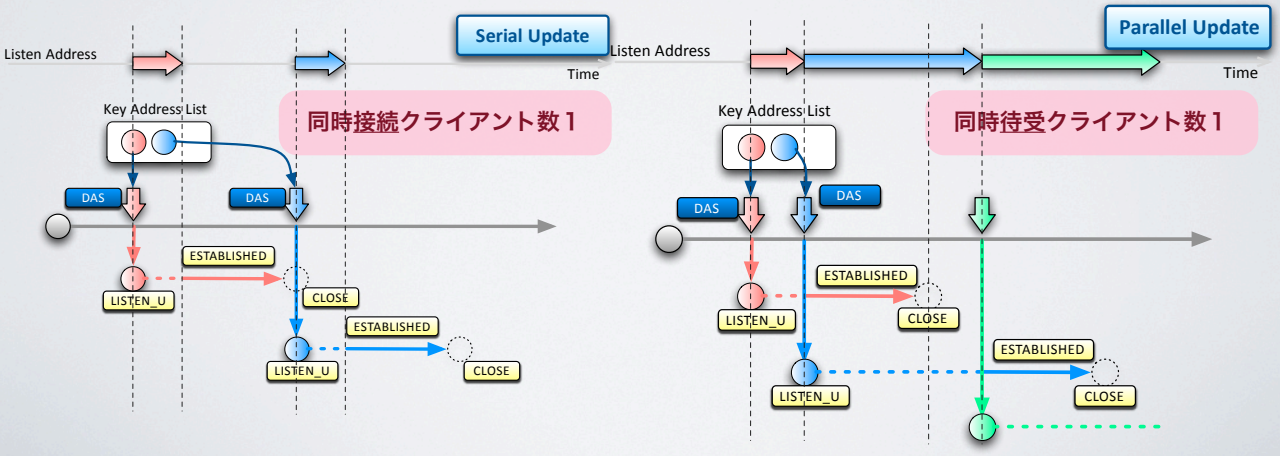
動作シーケンス

- **概要**
  - DAS (Delayed Address Setting) によるサーバソケットの待ち受けアドレスの設定と待ち受け
  - getaddrinfo によるサーバアドレスの取得と接続
  - 接続終了後にアドレスの破棄とアドレスリストの遷移



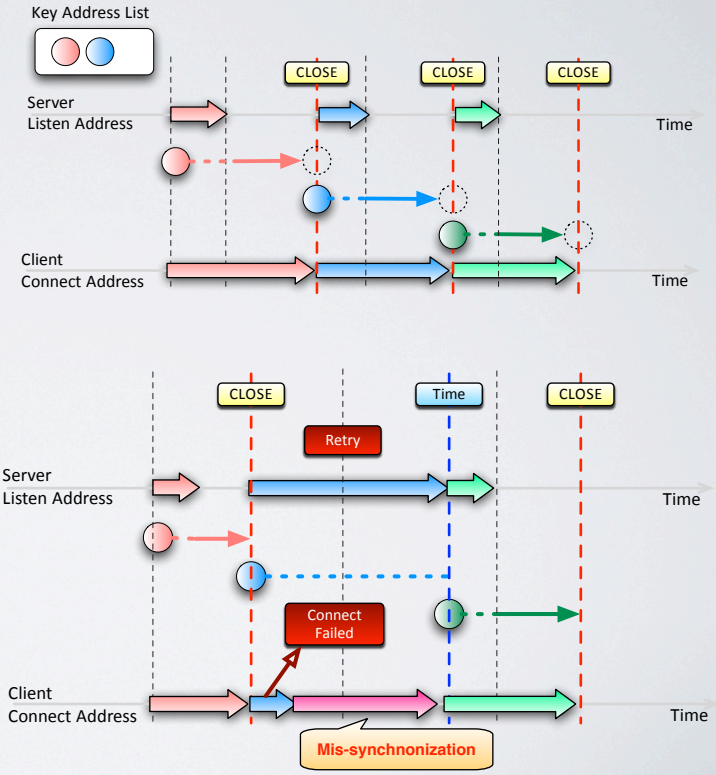
サーバ待ち受けアドレスの更新

- **シリアル更新型**
  - セッション終了後に新規待ち受け
- **パラレル更新型**
  - 接続完了後に新規待ち受け



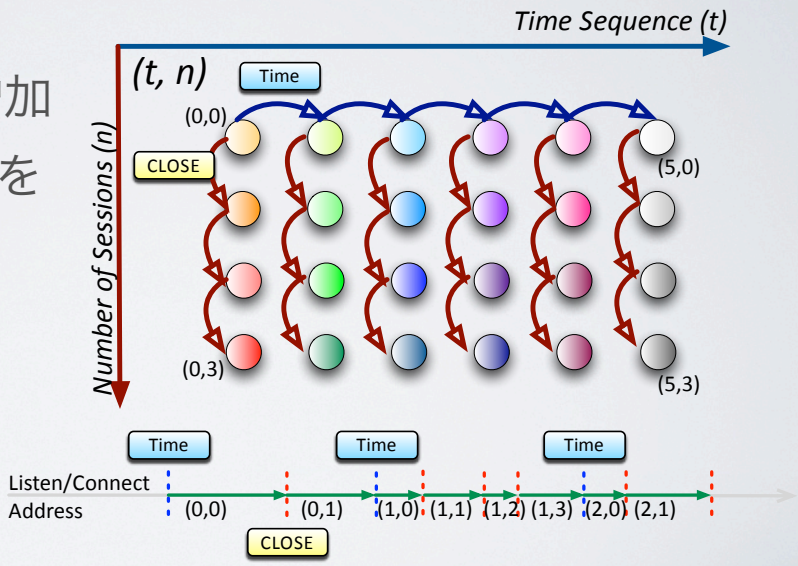
時刻情報を用いた再同期

- **「同期ずれ」の発生**
  - 機器の再起動
  - 複数端末による接続
  - 予期しない接続
- **時刻による再同期**
  - ある時刻間隔ごとにアドレスリストの再同期を行う



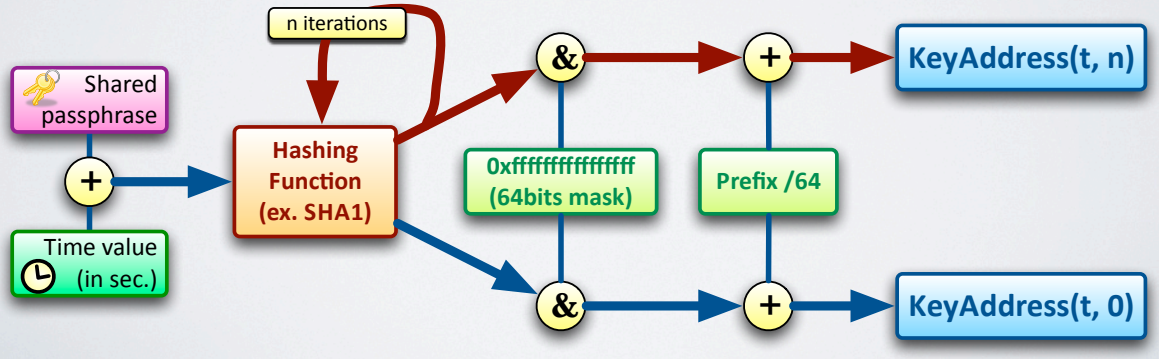
アドレス生成系列

- **「接続情報」「時刻情報」の二次元リスト**
  - (t, n): 時刻シーケンス t の n 番目のセッションの待ち受けアドレス
  - 接続ごとに n を増加
  - 時間同期ごとに t を増加して n を 0 に



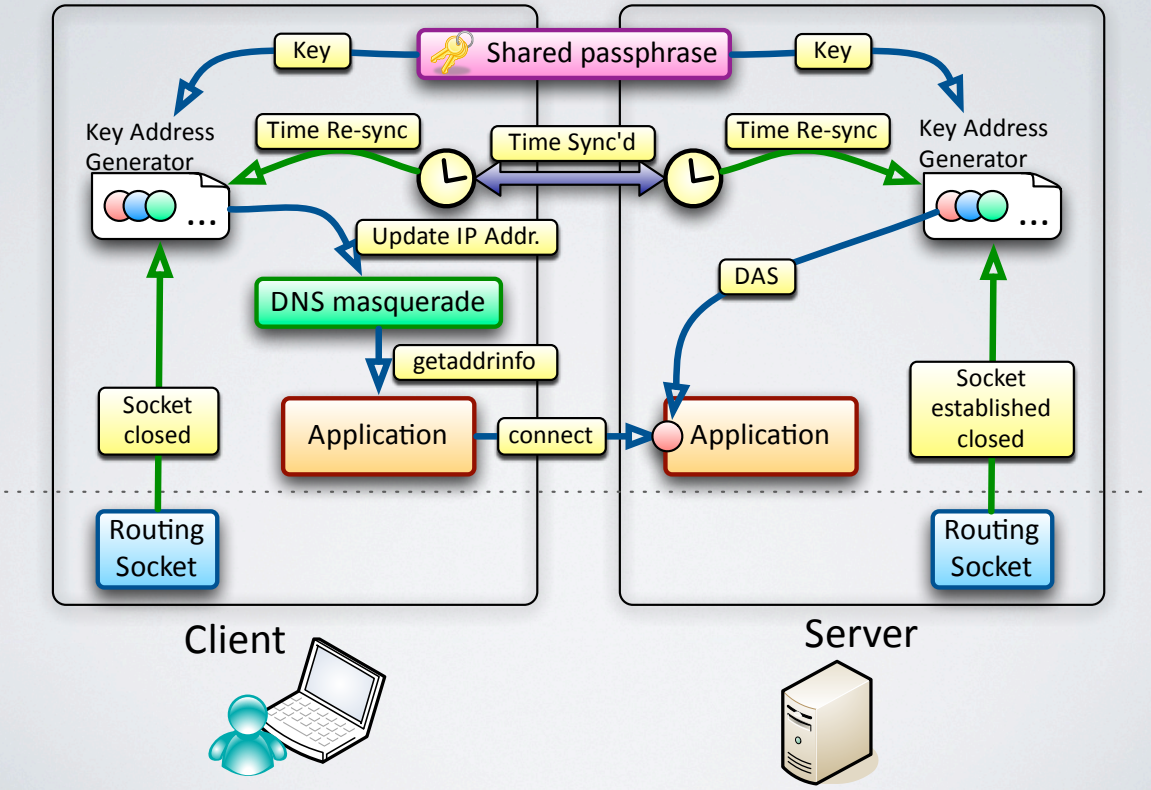
アドレス生成系列の実装例

- **共有パスフレーズと時刻情報をシード**
  - SHA1 (or SHA256) によるハッシュ化
  - 接続ごとに繰り返し計算
- **64ビット抽出しプレフィックスを付与**
  - ホスト ID 部分のみ変化



17

実装モデル



18

まとめと今後の課題

- **セッションごとに変化するアドレスによる安全な通信手法**
  - 誰もが実感できるセキュリティの実現による安全（＋安心感）の提供
  - 自己完結型同期によるアドレス共有・更新手法
    - エンドノード以外の実装が不要
- **今後の課題**
  - 実証実験を通じた安全性の確認
  - パラメータチューニング